**Amendments to the Claims:**

This listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. (Currently amended) A method in a data processing system for autonomically determining execution flow of a computer program, comprising:

providing a set of hardware registers for identifying a work area for a thread of the computer program, wherein the work area stores thread tracking information for the thread;

determining whether an overflow is about to occur in the work area;

responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers;

retrieving symbolic data for the thread, wherein retrieving symbolic data for the thread includes retrieving symbolic data from an indexed symbolic database by searching the indexed symbolic database for symbolic data based on a process identifier for the thread; and

generating a call sequence of the computer program based on the symbolic data for the thread.

2. (Original) The method of claim 1, wherein the set of hardware registers includes a work area register, a work area length register, and a current pointer register.

3. (Original) The method of claim 2, wherein the work area register includes a pointer pointing to a beginning of the work area for the thread.

4. (Original) The method of claim 2, wherein the work area length register includes one of a size of the work area for the thread or a pointer pointing to an end of the work area for the thread.

5. (Previously Presented) The method of claim 2, wherein the current pointer register includes a pointer pointing to a location of the work area where last thread tracking information is written.

6. (Original) The method of claim 1, wherein the thread tracking information for the thread includes a plurality of call stack entries for the thread.

7. (Original) The method of claim 6, wherein each of the plurality of call stack entries is written upon detection of one of a method call and a method return for the thread.

8. (Original) The method of claim 7, wherein each of the plurality of call stack entries includes an address to and an address from which one of the method call and a method return is executed.

9. (Original) The method of claim 8, wherein each of the plurality of call stack entries further includes additional information, and wherein the additional information includes time stamps and performance monitoring counter values.

10. (Original) The method of claim 9, wherein the additional information is compressed with the address to and the address from which one of the method call and a method return is executed when each of the plurality of call stack entries is written.

11. (Previously Presented) The method of claim 9, wherein the additional information is compressed with the address to and the address from which one of the method call and a method return is executed when the thread tracking information is copied from the work area to a buffer.

12. (Canceled)

13. (Previously Presented) The method of claim 1, wherein the buffer is one of a trace buffer and a consolidated buffer accessible by an application.

14. (Canceled)

15. (Previously Presented) The method of claim 1, wherein the symbolic data matches the process identifier for the thread and the address of one of a method call and a method return for the thread.

16. (Previously Presented) The method of claim 1, wherein retrieving symbolic data for the thread includes retrieving symbolic data from one of a directory of the loaded module, a shadow directory, or a loaded module if an address of the loaded module on a disk is known.

17. (Previously Presented) The method of claim 1, wherein generating a call sequence of the computer program includes associating the retrieved symbolic data with the thread tracking information in the buffer.

18. (Currently amended) A data processing system for autonomically determining execution flow of a computer program, the data processing system comprising:

providing means for providing a set of hardware registers for identifying a work area for a thread of the computer program, wherein the work area stores thread tracking information for the thread;

determining means for determining whether an overflow is about to occur in the work area;

copying means, responsive to determining that an overflow is about to occur in the work area, for copying the thread tracking information from the work area to a buffer using the set of hardware registers;

retrieving means for retrieving symbolic data for the thread, wherein the retrieving means comprises searching means for searching an indexed symbolic database for symbolic data based on a process identifier for the thread; and

generating means for generating a call sequence of the computer program based on the symbolic data for the thread.

19. (Canceled)

20. (Previously Presented) The data processing system of claim 18, wherein the generating means comprises:

associating means for associating the retrieved symbolic data with the thread tracking information in the buffer.

21. (Currently amended) A computer program product, including [[in]] a computer ~~readable~~ recordable-type medium storing computer readable program code for determining execution flow of a computer program, the computer program product comprising:

first instructions for providing a set of hardware registers for identifying a work area for a thread of the computer program, wherein the work area stores thread tracking information for the thread;

second instructions for determining whether an overflow is about to occur in the work area;

~~second~~ third instructions, responsive to determining that an overflow is about to occur in the work area, for copying the thread tracking information from the work area to a buffer using the set of hardware registers;

[[third]] fourth instructions for retrieving symbolic data for the thread, wherein the [[third]] fourth instructions comprises sub-instructions for searching an indexed symbolic database for symbolic data based on a process identifier for the thread; and

~~fourth~~ fifth instructions for generating a call sequence of the computer program based on the symbolic data for the thread.

22. (Canceled)

23. (Currently amended) The computer program product of claim 21, wherein the ~~fourth~~ fifth instruction comprises:

sub-instructions for associating the retrieved symbolic data with the thread tracking information in the buffer.